# 统一登录平台设计方案

# 统一登录

## 统一登录的应用背景

随着企业业务的日益增加，企业会不断上线新的应用。几乎所有的这些应用

都需要使用到用户登录功能，只有用户登录成功后才能访问应用中的资源。而对于一个集团公司而言，无论有多少个子系统，登录的入口应该只有一个，并且用户体系在整个集团中也只有一份，这个入口就是统一登录。

实施统一登录平台的好处：

1. 实施企业统一登录平台，用户登录的逻辑代码只有一份，它运行在统一登录服务器中，以后开发的各个子系统也无须再重复开发用户登录功能，提高了开发人员的开发效率。
2. 实施企业统一登录平台，用户只需要在浏览器上登录任意一个子系统即可访问整个集团所有的子系统，而无须再次输入用户名、密码登录，这样极大地提高了用户的使用体验。
3. 实施企业统一登录平台，可以统一用户的认证和鉴权过程，提高企业应用的安全性。

## 统一登录的web实现

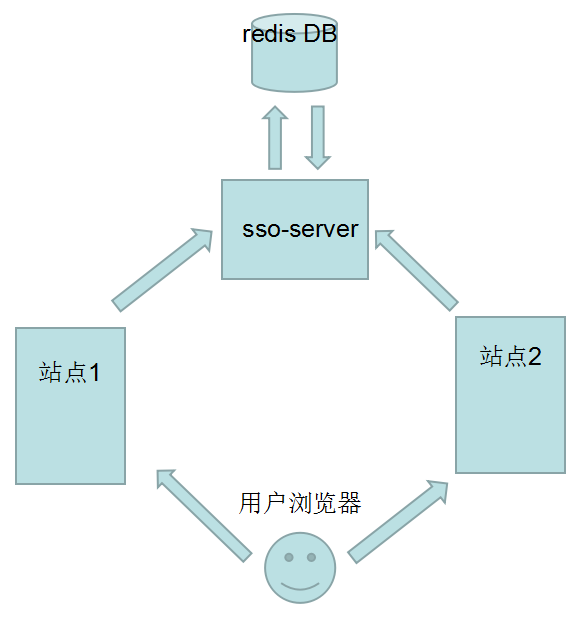
当用户浏览器访问网站时，浏览器会将网站的域和会话id保存到浏览器cookie中。如果网站没有做特别设置，cookie的生存周期与浏览器进程相同。即，当用户关闭浏览器时，浏览器进程结束，浏览器进程中的cookie信息也随之消失。当然，如果希望网站能永久记住当前用户的会话(凭证) ，在服务器上做特别的设置也是可以实现的。例如：在http 的response中设置cookie的有效期Integer.MAX\_VALUE，可避免浏览器关闭后cookie丢失的问题。

Cookie是一个很好的机制，它解决了http的无状态的问题，使得基于浏览器cookie机制的客户端能够很好地访问用户自身的资源。但是cookie也有一个明显的问题，即cookie不能跨域。如果用户希望在某系统站点A(www.site1.com)登录后，无需二次登录也可以访问与之关联的另一个站点B(www.site2.com)上受保护的资源，这就需要借助SSO了。

我们知道cookie机制决定了cookie是不可跨域的，对于这个问题我们不能从cookie本身上得到解决。但是我们可以借助SSO，将上述问题(一处登录，多处访问) 转到另一个角度并得到解决：提供登录的入口只有一个，即SSO，这样浏览器保存用户认证的cookie信息只来自一个域，即SSO服务器所在的域，具体实现步骤如下：

1. 对于浏览器访问的所有受保护的资源，子站点都将浏览器重定向到SSO进行登录验证(除非浏览器已在子站点登录)
2. SSO 服务器上提供登录页面，并验证用户提交的登录数据(用户名、密码等)，判断登录成功，创建SSO全局会话，再重定向到子站点指定的return\_url上。
3. 当用户退出时(在某个子站点退出)，子站点销毁局部会话，并重定向浏览器到SSO上，销毁全局会话，最后重定向到子站点指定的logout URL上。

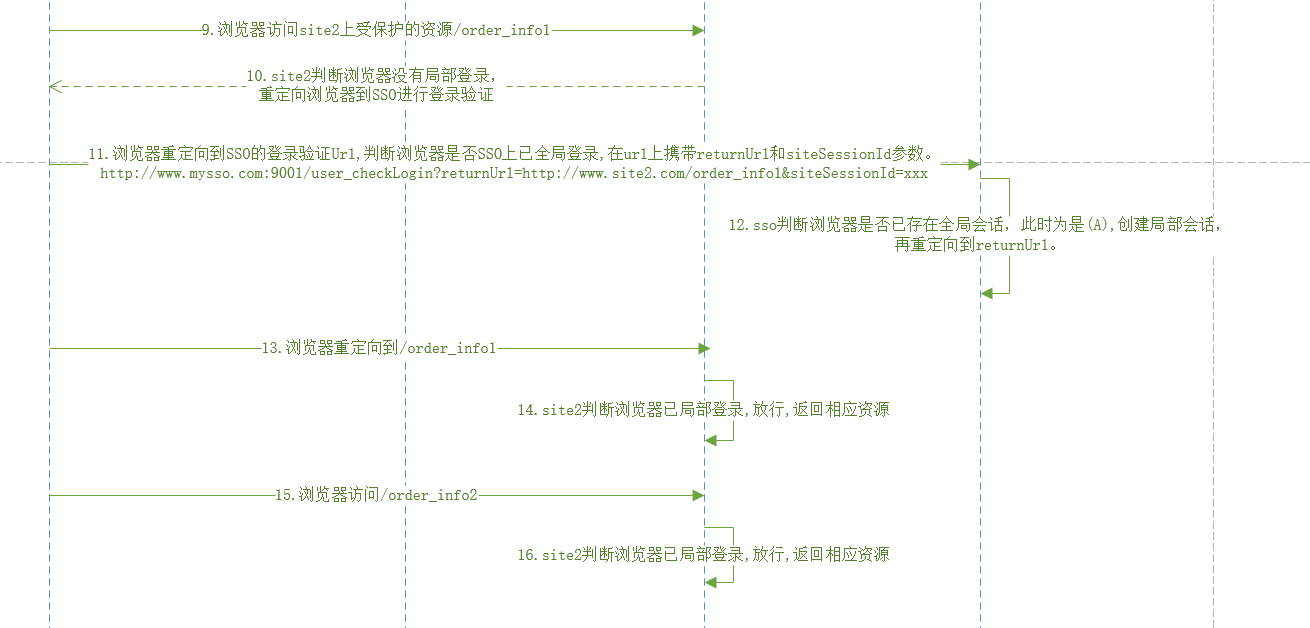
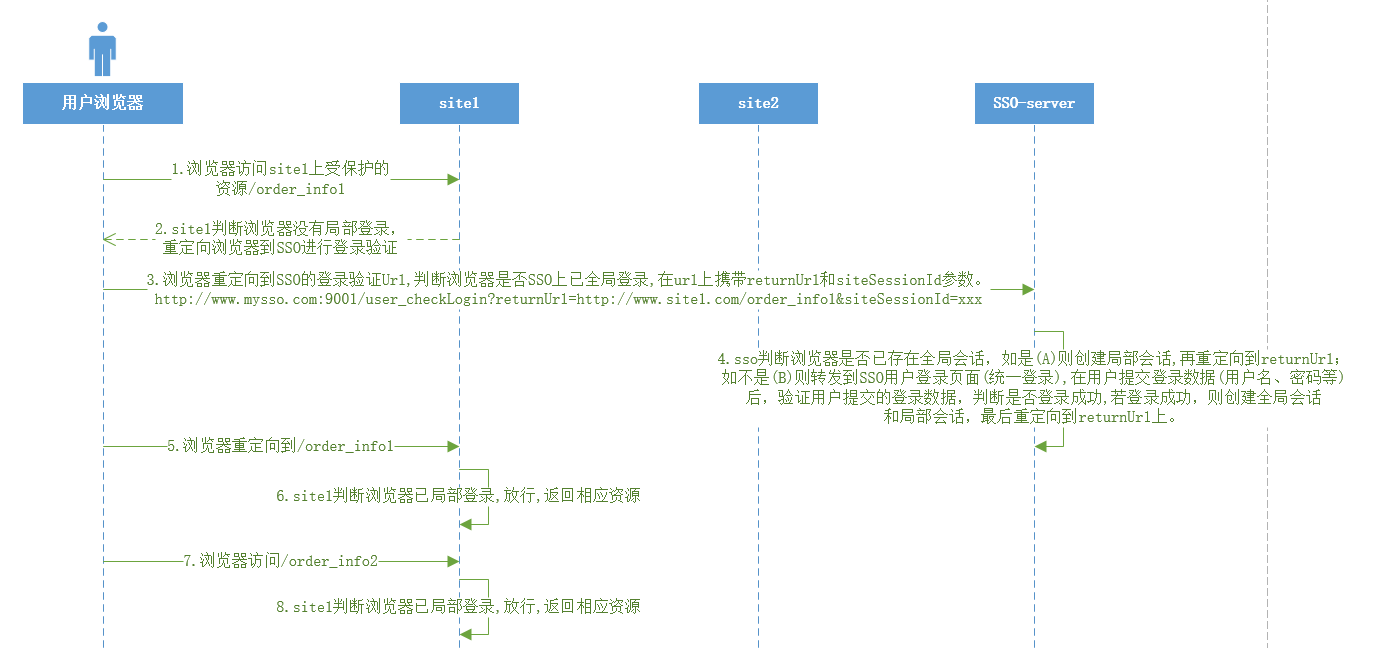
# 统一登录架构/流程图



为方便我们后续讨论，我们不妨做出如下假设：

1. 站点1请求url为http://www.site1.com:8001/
2. 站点2请求url为http://www.site2.com:8002/
3. sso-server请求url为http://www.mysso.com:9001/
4. 站点1上受保护的资源有/order\_info1 和/order\_info2
5. 站点2上受保护的资源有/order\_info1 和/order\_info2

## 2.1 统一登录流程



具体流程描述如下：

1. 浏览器访问site1上受保护的资源/order\_info1
2. site1判断浏览器没有局部登录，重定向浏览器到SSO进行登录验证
3. 浏览器重定向到SSO的登录验证Url,判断浏览器是否SSO上已全局登录,在url上携带returnUl和siteSessionId参数,即http://www.mysso.com:9001/

user\_checkLogin?returnUrl=http://www.site1.com/order\_info1&siteSessionId=xxx

1. sso判断浏览器是否已存在全局会话，如是(A)则创建局部会话,再重定向到returnUrl；

如不是(B)则转发到SSO用户登录页面(统一登录),在用户提交登录数据(用户名、密码等)

后，验证用户提交的登录数据，判断是否登录成功,若登录成功，则创建全局会话和局部会话，最后重定向到returnUrl上。

1. 浏览器重定向到/order\_info1
2. site1判断浏览器已局部登录,放行,返回相应资源
3. 浏览器访问/order\_info2
4. site1判断浏览器已局部登录,放行,返回相应资源
5. 浏览器访问site2上受保护的资源/order\_info1
6. site2判断浏览器没有局部登录，重定向浏览器到SSO进行登录验证
7. 浏览器重定向到SSO的登录验证Url,判断浏览器是否SSO上已全局登录,在url上携带returnUrl和siteSessionId参数。

[http://www.mysso.com:9001/user\_checkLogin?returnUrl=http://www.site2.com/order\_info1&siteSessionId=xxx](http://www.mysso.com:9001/user_checkLogin?returnUrl=http:/www.site2.com/order_info1&siteSessionId=xxx)

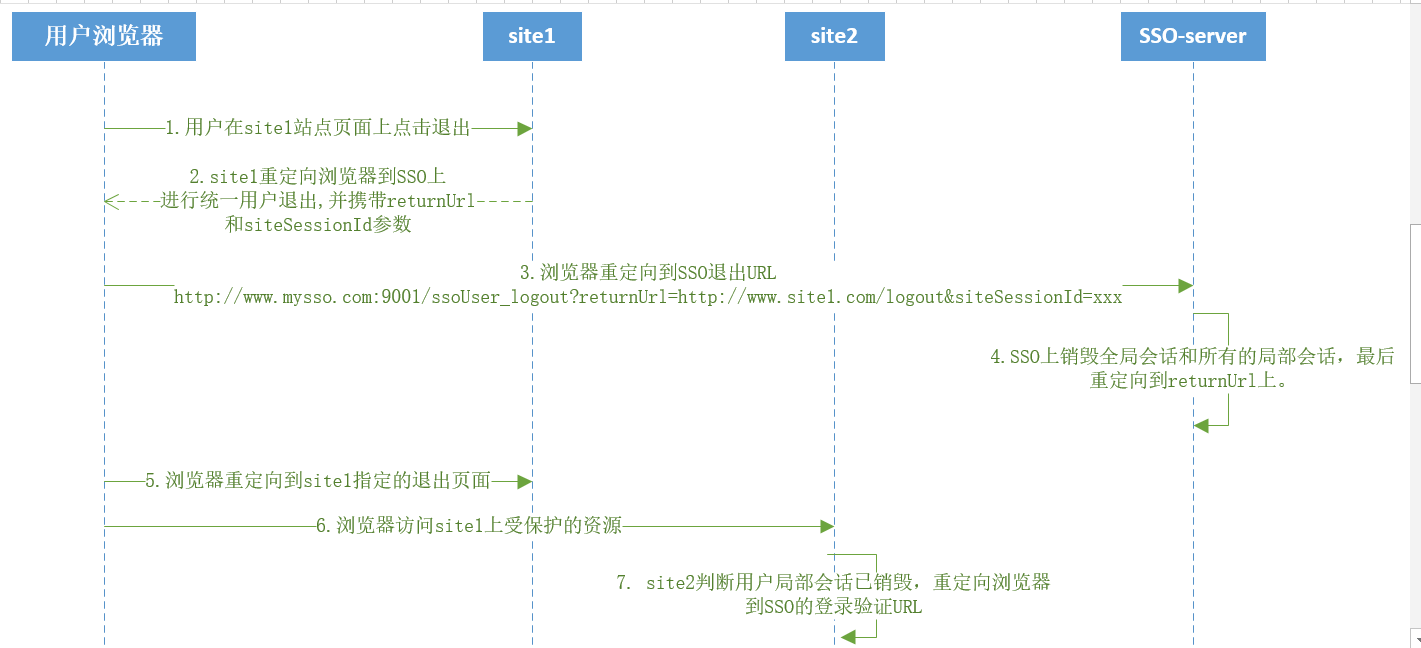
12. sso判断浏览器是否已存在全局会话，此时为是(A),创建局部会话，再重定向到returnUrl。

13. 浏览器重定向到/order\_info1

14. site2判断浏览器已局部登录,放行,返回相应资源

1. 浏览器访问/order\_info2
2. site2判断浏览器已局部登录,放行,返回相应资源

## 2.2 统一退出流程



具体流程描述如下：

1. 用户在site1站点页面上点击退出
2. site1重定向浏览器到SSO上进行统一用户退出,并携带returnUrl和siteSessionId参数
3. 浏览器重定向到SSO退出URL,即http://www.mysso.com:9001/ssoUser\_logout?

returnUrl=http://www.site1.com/logout&siteSessionId=xxx

1. SSO上销毁全局会话和所有的局部会话，最后重定向到returnUrl上。
2. 浏览器重定向到site1指定的退出页面
3. 浏览器访问site1上受保护的资源

7. site2判断用户局部会话已销毁，重定向浏览器到SSO的登录验证URL

# 子站点(springboot)如何接入

## 1、在子系统pom.xml中引入sso-client的插件包

<dependency>

<groupId>com.tingcream</groupId>

<artifactId>sso-starter-client</artifactId>

<version>1.0</version>

</dependency>

## 2、在application.yml中配置redis连接池及相应的sso信息

spring:

redis:

host: 192.168.100.220

port: 6379

password: 123456

ssl: **false**

database: 0

timeout: 6000

pool:

max-active: 8 #缺省值8

max-idle: 8 #缺省值8

min-idle: 0 #缺省值0

max-wait: 3000 #缺省值-1

sso:

ssoServer:

#sso 验证用户登录地址

checkLoginUrl: http://www.mysso.com:9001/ssoUser\_checkLogin

#sso 用户退出地址

logoutUrl: http://www.mysso.com:9001/ssoUser\_logout

ssoClient:

#本地退出回调页面

logoutPage: /logout

## 3、在springboot启动类上标记@EnableSsoAutoConfig注解，开启sso的自动化配置

@EnableSsoAutoConfig

**public** **class** SsoSite2Application **extends** SpringBootServletInitializer {

}

## 4、子站点如何获取统一登录(SSO)上登录的用户信息

@Autowired //注入bean

**private** SessionInfoHelper sessionInfoHelper;

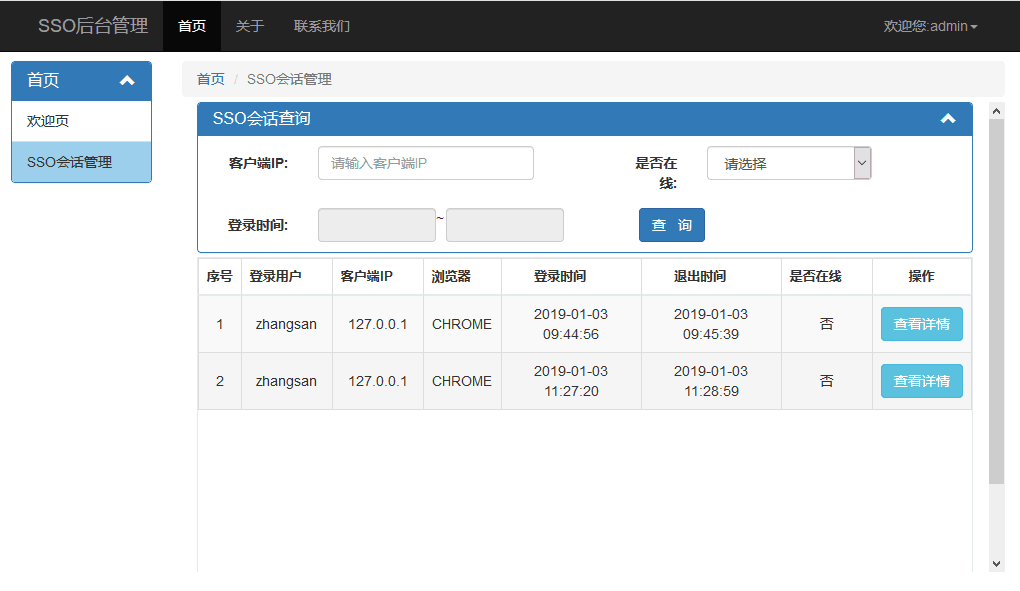
String sessionId = request.getSession().getId();

User sessionUser=sessionInfoHelper.getSessionUser(sessionId);

# 统一登录的后台设计

基于用户、角色、菜单的权限设计。

1. 控制某用户能访问某子站点(A、B)，而不能访问另一些子站点(C)。
2. 控制某用户能访问某子站点(A)的某些菜单，而不能访问站点的其他菜单。
3. 控制某用户能访问某子站点(A)的某些功能/按钮，而不能访问站点的其功能/按钮
4. 登录会话管理，如查看前台用户登录了哪些子系统，或者让前台用户强制下线等。



# 附录

[](统一登录与退出流程.vsdx)